SECTION A

objectives

In this section you will learn:

- About the World Wide Web
- What JavaScript is used for
- About Hypertext Markup Language
- How to create an HTML document
- About the JavaScript programming language
- About logic and debugging

# Programming, HTML, and JavaScript

## The World Wide Web

JavaScript lives and works within Web pages on the World Wide Web. To understand how JavaScript functions, it helps to know a little about how the World Wide Web operates.

The **World Wide Web** (the "Web") was created in 1989 at the European Laboratory for Particle Physics in Geneva, Switzerland, as a way to easily access cross-referenced documents that exist on the Internet. Documents are located and opened using **hypertext links**, which contain a reference to a specific document. **Hypertext Markup Language** (**HTML**) is a simple language used to design the Web pages that appear on the World Wide Web. A **Web browser** is a program that displays HTML documents on your computer screen. Currently, the two most popular Web browsers are Netscape Navigator and Microsoft Internet Explorer.


This textbook uses the terms HTML documents and Web pages interchangeably.

Every Web page or document has a unique address known as a **Uniform Resource Locator** (URL). You can think of a URL as a Web page's telephone number. Each URL consists of four parts: a protocol (usually HTTP), a service, either the domain

name for a Web server or a Web server's Internet Protocol address, and a file name. **Hypertext Transfer Protocol** (HTTP) manages the hypertext links that are used to navigate the Web; you can think of HTTP as driving the Web. HTTP ensures that Web browsers correctly process and display the various types of information contained in Web pages (text, graphics, and other information). The protocol portion of a URL is followed by a colon, two forward slashes, and the service, which is usually *www* for "World Wide Web." A **domain name** is a unique address used for identifying a computer, often a Web server, on the Internet. The domain name consists of two parts separated by a period. The first part of a domain name is usually composed of text that easily identifies a person or an organization, such as DonGosselin or Course. The last part of a domain name identifies the type of institution or organization. For instance, com (for *company*) represents private companies, gov (for *government*) represents government agencies, and edu (for *educational*) represents educational institutions. For example, course.com is the domain name for Course Technology. Examples of entire URLs include http://www.DonGosselin.com and http://www.course.com.

**An Internet Protocol, or IP address, is another way to uniquely identify computers or devices connected to the Internet, using a series of four groups of numbers separated by periods. All Internet domain names are associated with a unique IP address.**

In a URL, a specific filename, or a combination of directories and a filename, can follow a domain name or IP address. If the URL does not specify a filename, the requesting Web server looks for a file named INDEX.HTML in the root or specified directory. Figure 1-1 points out the parts of a sample URL that opens an HTML document named `html_ref.html`.

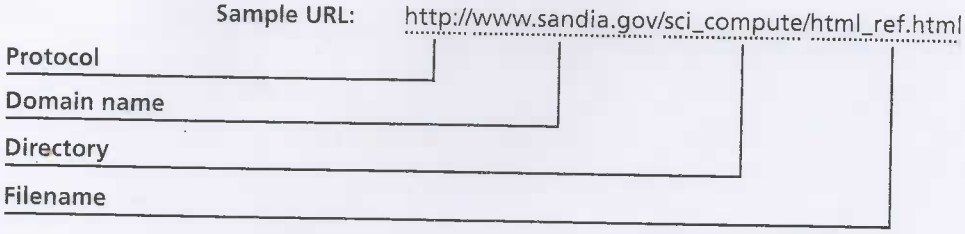



**Figure 1-1:** Sample URL

## JavaScript's Role on the Web

The original purpose of the World Wide Web was locating and displaying information. Once the Web grew beyond a small academic and scientific community, people began to recognize that greater interactivity would make the Web more useful. As commercial applications of the Web grew, the demand for more interactive and visually appealing Web sites also grew. Documents created using basic HTML, however, are static; the main purpose of HTML is to tell a browser how the document should appear. You can think of an HTML document as being approximately equivalent to a document created in a word-processing or desktop publishing program—the only thing you can do with it is view or print it. In response to the demand for greater interactivity, Netscape developed the JavaScript programming language for use in Navigator Web browsers.

JavaScript brings HTML to life and makes Web pages dynamic. Instead of HTML documents being static, JavaScript can turn them into applications, such as games or order forms. You can use JavaScript to change the contents of a Web page after it has been rendered by a browser, to interact with a user through forms and controls, to create visual effects such as animation, and to control the Web browser window itself. None of these things was possible before the creation of JavaScript.

Thanks in large part to JavaScript, the Web today is used for many different purposes, including advertising and entertainment. Many businesses today (and probably most in the future) have a Web site. To attract people to a Web site, and keep them there, a business's Web site must be exciting, interactive, and visually stimulating. Business Web sites use "flashing signs" that advertise specials, animation, interactivity, intuitive navigation controls, and many other types of effects to help sell their products. It is also easy to find games, animation, and other forms of entertainment on the Web. All ~~of~~ these types of applications and effects can be created with JavaScript.

To gain a better idea of what you can do with JavaScript, examine Figures 1-2, 1-3, and 1-4. You will be creating these programs in later tutorials. The map displayed in Figure 1-2 is an image map. Passing your mouse over a country on the map highlights the country and displays its name at the bottom of the Web page. You will create the image map in Tutorial 2. Figure 1-3 displays an online calculator you will create in Tutorial 3. The last program, in Figure 1-4, is an online product registration form that you will create in Tutorial 6. These are just some of the examples of the programs you will create with JavaScript in this textbook.

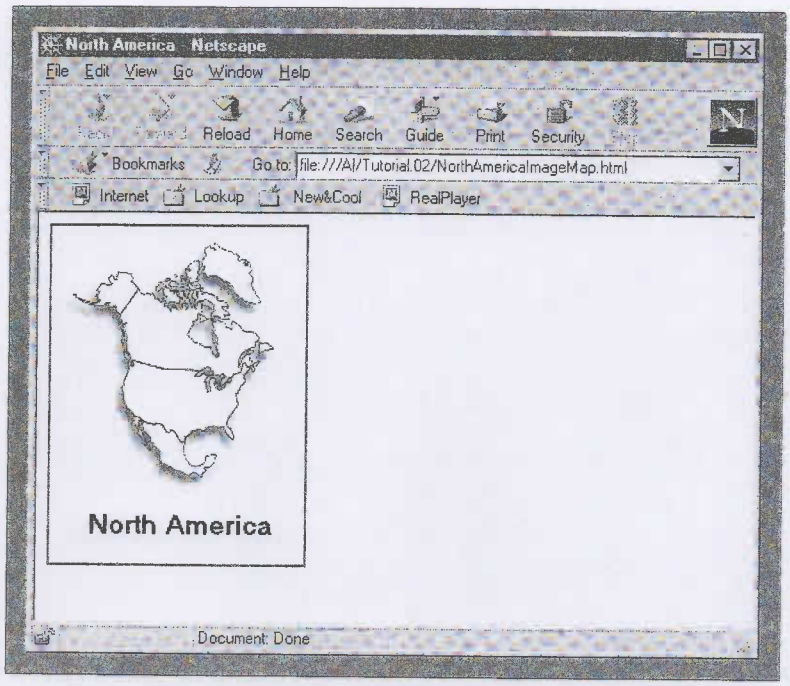



**Figure 1-2:** Image map

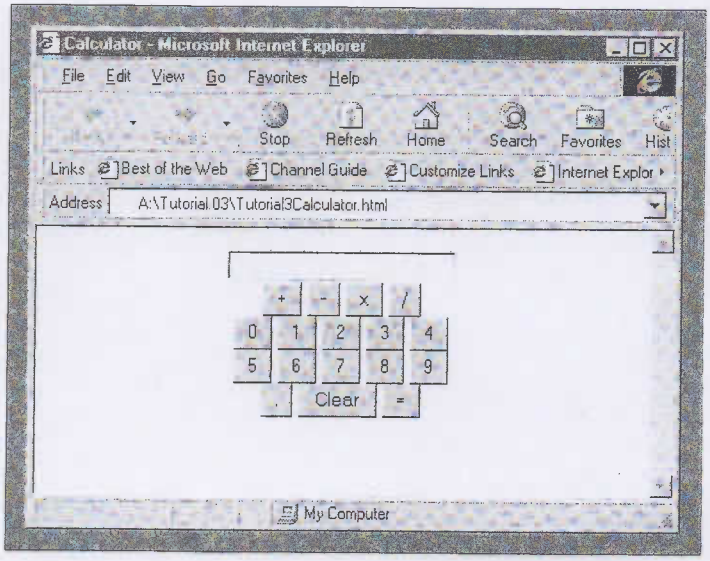



**Figure 1-3:** Online calculator

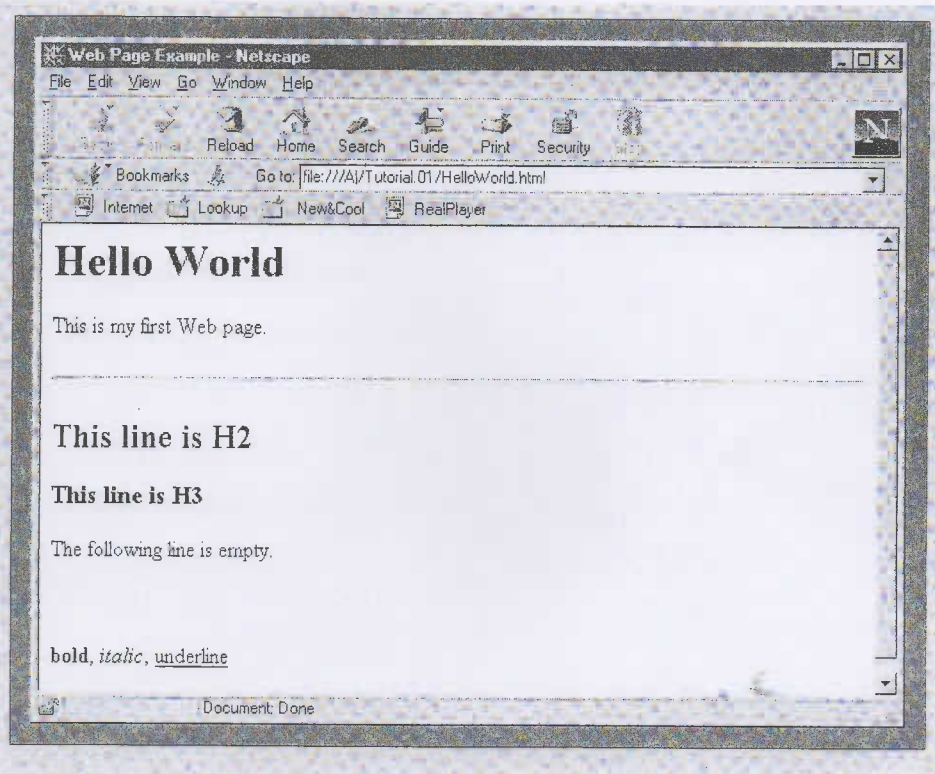



**Figure 1-10:** HelloWorld.html in Navigator 4.0

## The JavaScript Programming Language

JavaScript is a scripting language. The term **scripting language** refers to programming languages that are executed by an interpreter from within a Web browser. An **interpreter** translates programming code into an executable format *each time* the program is run—one line at a time. Programs written in scripting languages, such as JavaScript, are interpreted when a scripting engine loads an HTML page. A **scripting engine** is an interpreter that is part of the Web browser. A Web browser that contains a scripting engine to translate scripts is called a **scripting host**. Navigator and Internet Explorer are both examples of scripting hosts for JavaScript programs.

The JavaScript language was first introduced in Navigator and was originally called LiveScript. With the release of Navigator 2.0, the name was changed to JavaScript 1.0. Subsequently, Microsoft released its own version of JavaScript in Internet Explorer 4.0 and named it JScript. The most current versions of each implementation are JavaScript 1.3 in Navigator and JScript 3.0 in Internet Explorer. However, these two implementations are not entirely compatible with each other. The goal of this text is to create JavaScript programs that can run on either Navigator or Internet Explorer. Therefore, this text focuses on JavaScript

code that is compatible with Netscape JavaScript 1.2, which is supported by Navigator 4 and Internet Explorer 4. Netscape JavaScript 1.2 is considered the current JavaScript standard and is roughly compatible with JScript 2.0.

**The European Computer Manufacturer's Association, or ECMA, produced an international, standardized version of JavaScript called ECMAScript. ECMAScript is roughly compatible with JavaScript 1.1. Future versions of Netscape JavaScript are expected to completely conform to ECMAScript. Microsoft claims that JScript 3.0 is already 100% compliant with ECMAScript.**

**Many people think that JavaScript is related to or is a simplified version of the Java programming language. They are, however, entirely different languages. Java is a compiled, object-oriented programming language that was created by Sun Microsystems and is considerably more difficult to master than JavaScript. JavaScript is an interpreted scripting language that was created by Netscape. Although Java is often used to create programs that can run from a Web page, Java programs are external programs that execute independently of a browser. In contrast, JavaScript programs run within a Web page and control the browser.**

JavaScript is available in two formats: client-side JavaScript and server-side JavaScript. The standardized **client-side JavaScript** is the format available to HTML pages displayed in Web browsers (the *client*). JavaScript version 1.2 in Navigator 4.0 and ECMAScript are client-side versions of JavaScript. Server-side JavaScript is used with Web servers to access file systems, communicate with other applications, access databases, and perform other tasks. Currently, server-side JavaScript is proprietary and vendor-specific. You must know a slightly different version of the language for each vendor's Web server; there is no server-side standard similar to ECMAScript. Client-side and server-side JavaScript share the same basic programming features. Figure 1-11 illustrates how client-side and server-side JavaScript are related.

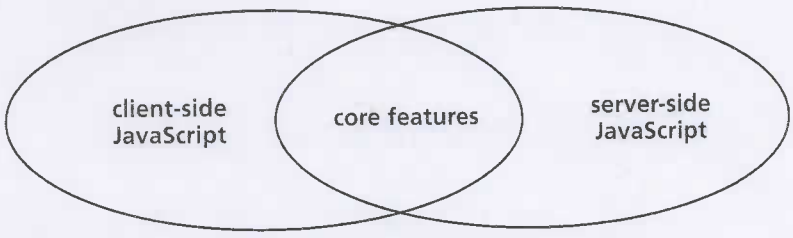



**Figure 1-11:** Relationship of client-side and server-side JavaScript

## Logic and Debugging

All programming languages, including JavaScript, have their own **syntax**, or rules of the language. To write a program, you must understand a given programming language's syntax. You must also understand computer programming logic. The **logic** underlying any program involves executing the various parts of the program in the correct order to produce the desired results. For example, although you know

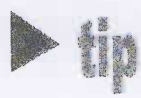
how to drive a car well, you may not reach your destination if you do not follow the correct route. Similarly, you might be able to use a programming language's syntax correctly, but be unable to execute a logically constructed, workable program. Examples of logical errors include multiplying two values when you meant to divide them, or producing output prior to obtaining the appropriate input. The following JavaScript code contains another example of a logic error:

```
var count = 1
while (count <= 10) {
     alert("The number is " + count);
}
```

The code in the example uses a while statement, which is used to repeat a command or series of commands based on the evaluation of certain criteria. The criterion in the example is the value of a variable named count (you will learn about variables in Tutorial 2). The while statement is supposed to execute until the count variable is less than or equal to 10. However, there is no code within the while statement's body that changes the count variable's value. The count variable will continue to have a value of 1 through each iteration of the loop. In this case, an alert dialog box containing the text string *The number is 1* will be displayed over and over again, no matter how many times you press the OK button. This type of logical error is called an infinite loop.

**Do not worry about how the JavaScript code in the example is constructed. The example is only meant to give you a better understanding of a logical error.**

Any error in a program that causes it to function incorrectly, whether due to incorrect syntax or flaws in logic, is called a bug. **Debugging** describes the act of tracing and resolving errors in a program. Legend has it that the term *debugging* was first coined by Grace Murray Hopper, a mathematician who was instrumental in developing the COBOL programming language. As the story from the 1940s goes, a moth short-circuited a primitive computer that Hopper was using. Removing the moth from the computer *debugged* the system and resolved the problem. Today, a bug refers to any sort of problem in the design and operation of a program.



**Do not confuse bugs with computer viruses. Bugs are problems within a program that occur because of syntax errors, design flaws, or run-time errors. Viruses are self-contained programs designed to "infect" a computer system and cause mischievous or malicious damage. Actually, virus programs themselves can contain bugs if they contain syntax errors or do not perform (or do damage) as their creators envisioned.**

Many programming languages include commands and other features to assist in locating bugs in a program. However, the version of JavaScript covered in this textbook does not contain any useful debugging tools. Future versions of JavaScript are expected to include debugging features. To provide you with some debugging skills, debugging suggestions are offered in the Tips feature throughout this book.

However, as you read the debugging tips, keep in mind that debugging is not an exact science—every program you write is different and requires different methods of debugging. Your own logical and analytical skills are the best debugging resources you have.

In the next section you will start learning how to create JavaScript programs.

## SUMMARY

- Hypertext Markup Language (HTML) is a simple protocol used to design Web pages that appear on the World Wide Web.
- The World Wide Web is driven by Hypertext Transfer Protocol (HTTP), which manages the hypertext links that are used to navigate the Web.
- Every Web document has a unique address known as a Uniform Resource Locator (URL).
- JavaScript brings HTML to life and makes Web pages dynamic, turning them into applications, such as games or order forms.
- HTML documents must be text documents that contain formatting instructions, called **tags**, along with the text that is to be displayed on a Web page.
- HTML tags range from formatting commands to controls that allow user input, to tags that allow the display of graphic images and other objects.
- A Web browser's process of assembling and formatting an HTML document is called parsing or rendering.
- An interpreter translates programming code into an executable format each time the program is run—one line at a time.
- JavaScript is an interpreted programming language.
- A scripting engine is an interpreter that is part of the Web browser. A Web browser that contains a scripting engine to translate scripts is called a scripting host.
- Standardized client-side JavaScript is the JavaScript format available to HTML pages displayed in Web browsers.
- All programming languages, including JavaScript, have their own syntax, or rules of the language.
- The logic behind any program involves executing the various parts of the program in the correct order to produce the desired results.
- Debugging describes the act of tracing and resolving errors in a program.

Fonte : Gosselin, D. 2000. JavaScript. Course Technology, Thompson Learning